

Programa del curso IC-4700

## **Lenguajes de Programación**

Escuela de Computación  
Carrera de Ingeniería en Computación, Plan 412.

## I parte: Aspectos relativos al plan de estudios

### 1 Datos generales

<b>Nombre del curso:</b>	Lenguajes de Programación
<b>Código:</b>	IC-4700
<b>Tipo de curso:</b>	Teórico-Práctico
<b>Electivo o no:</b>	No
<b>Nº de créditos:</b>	4
<b>Nº horas de clase por semana:</b>	4
<b>Nº horas extraclasses por semana:</b>	8
<b>Ubicación en el plan de estudios:</b>	Curso del IV Semestre del Bachillerato de Ingeniería en Computación
<b>Requisitos:</b>	IC-3002 Análisis de Algoritmos IC-3101 Arquitectura de Computadoras
<b>Correquisitos:</b>	Ninguno.
<b>El curso es requisito de:</b>	IC5701 Compiladores e Intérpretes
<b>Asistencia:</b>	Obligatoria
<b>Suficiencia:</b>	No
<b>Posibilidad de reconocimiento:</b>	Sí
<b>Vigencia del programa:</b>	I Semestre de 2018

## **2 Descripción general**

En este curso se estudian con detalle los paradigmas fundamentales de los lenguajes de programación. Se aprenden lenguajes representativos de cada paradigma y se estudia la teoría más relevante del diseño de lenguajes

## **3 Objetivos**

### **Objetivo General**

Estudiar los conceptos fundamentales y los principales principios de diseño de los lenguajes de programación. Estudiar los cuatro paradigmas principales de programación y ofrecer criterios para comparar, evaluar y seleccionar lenguajes de programación en relación con sus diferentes usos.

### **Objetivos Específicos**

1. Aprender un lenguaje ejemplo para cada uno de los cuatro paradigmas principales de programación, así como conocer varios otros lenguajes para representar conceptos que no tengan los lenguajes ejemplo.
2. Comprender algunos principios generales de diseño de lenguajes.
3. Ofrecer criterios para comparar, evaluar y seleccionar lenguajes de programación.
4. Dar criterios para discernir cuál es el lenguaje de programación más adecuado para un problema dado

## **4 Contenidos**

### **1. Introducción y conceptos fundamentales.**

- 1.1. Evolución histórica de los lenguajes de programación.
- 1.2. Principios de diseño de lenguajes de programación.
- 1.3. Sintaxis, semántica y pragmática.
- 1.4. Conceptos fundamentales
  - 1.4.1. Valores, datos y tipos.
- 1.5. Expresiones.
- 1.6. Almacenamiento y control.
- 1.7. Asociación ("binding").
- 1.8. Abstracción.
- 1.9. Encapsulación.
- 1.10. Secuenciadores.

### **2. Programación imperativa. [Lenguajes: C o Ada o CLU]**

- 2.1. Modelos de almacenamiento (punteros, arreglos, etc.)
- 2.2. Mecanismo de paso de parámetros

- 2.3. Portabilidad
- 2.4. Preprocesamiento, entrada y salida estándar
- 2.5. Modularidad
- 2.6. Generalidad
- 2.7. Ocultamiento de información
- 2.8. Independencia de representación
- 2.9. Referencia a los conceptos fundamentales

**3. Programación funcional. [Lenguaje: preferiblemente Standard ML y Haskell. Otros lenguajes: Lisp, Scheme]**

- 3.1. Expresiones y aritmética
- 3.2. Funciones como "ciudadanos de primera clase"
- 3.3. Manejo de listas como primitivas
- 3.4. Principios de diseño y programación funcional
- 3.5. Streams (evaluación perezosa)
- 3.6. Programación funcional con tipos
- 3.7. Polimorfismo paramétrico

**4. Programación lógica. [Lenguaje: Prolog o Turbo Prolog. Otros lenguajes ilustrativos: Oz o Gödel]**

- 4.1. Relaciones vs funciones
- 4.2. Hechos y consultas
- 4.3. Calculo de predicados
- 4.4. Dominios, datos compuestos y listas.
- 4.5. Unificación.
- 4.6. Control de flujo
  - 4.6.1. Backtracking y orden de descripción.
- 4.7. Corte y Fail.

**5. Programación orientada a objetos. [Lenguaje Smalltalk, Eiffel, Java. Otros lenguajes ilustrativos: BETA, C++, Object Pascal (Delphi)]**

- 5.1. Objetos y Mensajes
- 5.2. Expresiones y aritmética
- 5.3. Clases y Métodos
- 5.4. Instancia y tipos de variables
- 5.5. Herencia y polimorfismo
- 5.6. Jerarquía de clases
- 5.7. Colecciones
- 5.8. Principios de diseño y programación OO
- 5.9. Bloques de código y mensajes en cascada

5.10. Estudio analítico de otros lenguajes [lenguaje ilustrativo: Java]

## **6. Elementos avanzados de lenguajes de programación**

6.1. Concurrencia, paralelismo y distribución.

6.1.1. [Modelo de concurrencia de Java]

6.1.2. [Herramientas para programación máquinas multinúcleo]

6.1.3. [Lenguajes: occaml, POOL, C-Linda, Orca].

6.2. Sistemas de tipos.

6.3. Ligas entre programas de distintos lenguajes.

6.4. Elementos del diseño de lenguajes de programación.

6.5. Evaluación y selección de lenguajes de programación

## **II parte: Aspectos operativos**

### **5 Metodología de enseñanza y aprendizaje**

Los estudiantes harán programas no triviales para cada paradigma fundamental (imperativo, orientado a objetos, funcional y lógico)

Para cada uno de los paradigmas fundamentales se usará un lenguaje principal como ejemplo de estudio. Se ilustrarán variante del paradigma con lenguajes afines.

### **6 Evaluación**

<b>Rubro</b>	<b>Porcentaje</b>
Exámenes	25%
Lecturas	15%
Proyectos programados	60%
Total	100%

### **7 Bibliografía**

**Obligatoria**

Barendregt, H. (1985). *The Lambda Calculus: its Syntax and Semantics*. North-Holland, Amsterdam.

*Turbo Prolog Reference Guide V 2.0*. (1988). Borland International.

Camacho, L. *Lógica Simbólica*. Cartago, Costa Rica: Editorial Tecnológica.

Copi, I. (1997). *Lógica Simbólica*. Buenos Aires, Argentina: EUDEBA.

Inc., D. (1986). *Smalltalk V: Tutorial and programming Handbook*. Digitalk Inc.

Friedman, D. (1992). *From Babbage to Babel and beyond: A brief history of programming languages*. Computer Language 17.

Friedman, D., & Felleisen, M. (1999). *The Little Schemer* (4ta edición ed.). Science Research Associates Inc, SRA Inc, The MIT Press.

Helo, J. (2000). *Introducción a la Programación con Scheme*. Cartago, Costa Rica: Editorial Tecnológica de Costa Rica.

Louden, K. (2004). *Lenguajes de Programación* (2da Edición ed.). Thomson.

Hoare. (1973). *Hints on programming language design*. Hoare y Jones.

*Essays in Computing Science*. (1989). Prentice-Hall.

Horowitz. *Programming languages, a grand tour*. Computer Science Press / Springer-Verlag.

Gordon, M. (1988). *Programming Language Theory and its implementation*. New York: Prentice-Hall.

Kernighan, B., & Ritchie, D. (1985). *El lenguaje de programación C*. Prentice Hall Hispanoamericana.

Lennan, B. (1986 (1995)). *Principles of Programming Languages: Design, Evaluation, and Implementation* (2nd. edition ed.). Holt, Rinehart & Winston (ahora Oxford University Press).

Pratt, T., & Zelkowitz, M. (1998). *Lenguajes de programación: diseño e implementación*. Prentice-Hall Hispanoamericana.

Sethi, R. (1992). *Lenguajes de programación: conceptos y constructores*. Addison-Wesley.

Stansifer. (1995). *The study of programming languages*. Prentice-Hall.

Tennent. (1981). *Principles of programming languages*. Prentice-Hall.

Watt, D. (2004). *Programming language design concepts*. John Wiley & Sons.

Wirth. (1974). *On the design of programming languages*. Horowitz.

*Programming languages, a grand tour*. Computer Science Press / Springer-Verlag.

### **Adicional**

No tiene Bibliografía adicional.

## **8 Profesor**